ХРАНЕНИЕ УЧЕТНЫХ ЗАПИСЕЙ В ANDROID

Костюк А. С.

УО «Гродненский государственный университет им. Я. Купалы», Гродно, Беларусь,
e-mail: andrei.kastsiuk@gmail.com

Одно из направлений, обслуживаемое с помощью мобильных приложений, - это упрощенный вариант работы с группой веб-сервисов, требующих аутентификации. Поэтому при разработке, в частности, Android-приложений необходимо решить задачу об организации хранения учетных записей пользователя. Достаточно трудозатратна разработка собственного механизма хранения данных об учетных записях на устройстве пользователя в файлах SharedPreferences или в базах данных SQLLite.

В Android существует Account Manager для управления учетными записями в системе, который позволяет хранить любые данные об этих записях на устройстве пользователя, поддерживать различные права доступа и использовать для аутентификации в различных сервисах.

Создание своей учетной записи и механизма аутентификации требует реализации класса Authenticator'a, Activity с разметкой входа, Service'a интеграции в систему и указания в манифесте приложения следующих разрешений: MANAGE_ACCOUNTS, AUTHENTICATE_ACCOUNTS, USE_CREDENTIALS. Для создания класса Authenticator нужно расширить класс AbstractAccountAuthenticator и создать xml-файл с описанием этого Authenticator'a (название, иконка, тип и др.). Ключевыми являются методы addAccount() и getAuthToken(). Метод addAccount() вызвается при попытке добавления учетной записи в систему и в качестве результата возвращает Activity для аутентификации. С помощью метода getAuthToken() можно получить программный токен, который будет храниться в ключе KEY_AUTHTOKEN. Реализовать процесс получения программного токена от сервера можно любым удобным для разработчика способом, например, используя класс AsyncTaskLoader. Activity для аутентификации должна быть унаследована от класса AccountAuthenticatorActivity. После успешного получения программного токена, учетную запись можно добавить на устройство с помощью метода addAccountExplicitly(). Для передачи результата с данными в Account Manager, используется метод setAccountAuthenticatorResult(). Для того чтобы Authenticator был доступен системе и мог использоваться приложениями, необходимо создать сервис интеграции, такой сервис будет работать в фоновом режиме. Для этого следует создать экземпляр Authenticator'a в методе сервиса onCreate() и вернуть интерфейс IBinder этого экземпляра с помощью метода getIBinder() в методе onBind(). Чтобы сервис был доступен в системе, нужно указать его в файле манифеста приложения.

После запуска приложения и успешной аутентификации, создается собственная учетная запись, которой можно пользоваться без повторного взаимодействия со стороны пользователя с формами входа в сервисы.

Литература